BatLab Basic Project Kit – Potentiometer – Dimming an LED (analogWrite)

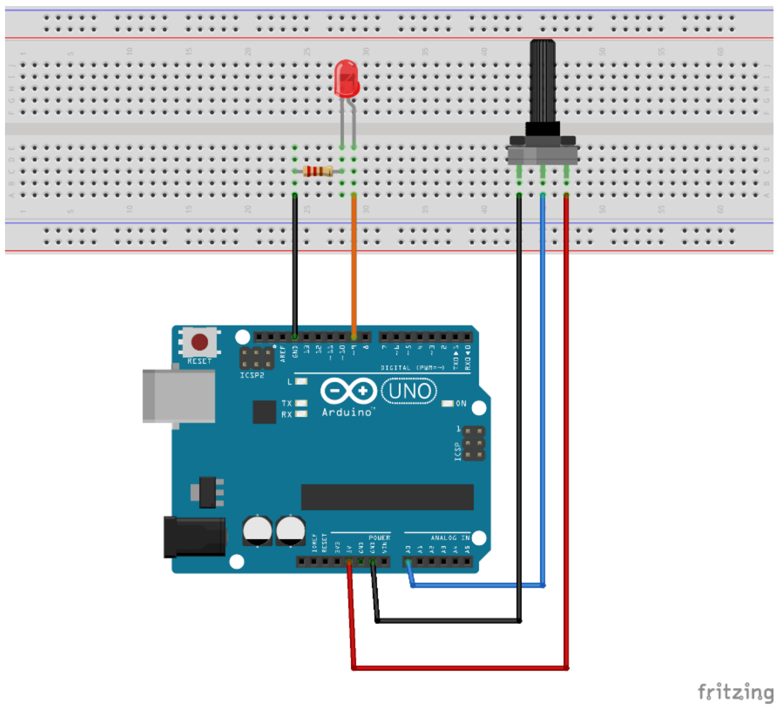
HOW IT WORKS

This experiment is wired exactly the same as the *Potentiometer – Blinking an LED* experiment, but now we are going to use a technique called "Pulse Width Modulation" or PWM to dim the LED…or at least, make it look dim to our eyes. For more technical information on PWM, see https://www.arduino.cc/en/Tutorial/PWM.

PARTS

- Arduino Uno, breadboard, and jumper wires
- Potentiometer (knob)
- LED (any color)
- 220Ω resistor (red marking)

CIRCUIT



CODE

```
/* POTENTIOMETER – DIMMING AN LED

Hardware connections:

  Potentiometer:

    Potentiometers have three pins. When we're using it as a
    voltage divider, we connect the outside pins to power and
    ground. The middle pin will be the signal (a voltage which
    varies from 0 Volts to 5 Volts depending on the position of
    the knob).

    Connect the middle pin to ANALOG IN pin 0 on the Arduino.
    Connect one of the outside pins to 5V.
    Connect the other outside pin to GND.

    (TIP: if once your program is running, the knob feels
    "backwards", you can swap the 5V and GND pins to reverse
    the direction.)

  LED:

  Connect the positive side of your LED (longer leg) to
   Arduino DIGITAL pin 9

  Connect the negative side of your LED (shorter leg) to a 220 Ohm
resistor

  Connect the other side of the resistor to ground.

This sketch is a modification of Circuit #6 from the
Sparkfun Inventor's Kit, written by Sparkfun Electronics.
http://learn.sparkfun.com/

It’s also similar to this sketch: Examples->Analog->AnalogInOutSerial

*/


// As usual, we'll create constants to name the pins we're using.
// This will make it easier to follow the code below.

const int sensorPin = 0;
const int ledPin = 9;

int potValue = 0; // for the reading from the potentiometer
int lightLevel = 0; //for the output to the LED
```

```
void setup()
{
  // Set up the LED pin to be an output.
  // (We don't need to do anything special to use the analog input.)

  pinMode(ledPin, OUTPUT);
}


void loop()
{

  potValue = analogRead(sensorPin);

  // We now want to use this number to control the brightness of
  // the LED. But we have a problem: the analogRead() function
  // returns values between 0 and 1023, and the analogWrite()
  // function wants values from 0 to 255.

  // We can solve this by using two handy functions called map()
  // and constrain(). Map will change one range of values into
  // another range. If we tell map() our "from" range is 0-1023,
  // and our "to" range is 0-255, map() will squeeze the larger
  // range into the smaller. (It can do this for any two ranges.)

  lightLevel = map(potValue, 0, 1023, 0, 255);

  analogWrite(ledPin, lightLevel);

// wait 2 milliseconds before the next loop
// for the analog-to-digital converter to settle
// after the last reading

   delay(2);

}
```